MUSIC GENRE CLASSIFICATION USING SIMILARITY FUNCTIONS

Yoko Anan, Kohei Hatano, Hideo Bannai and Masayuki Takeda
Department of Informatics, Kyushu University
{yoko.anan, hatano, bannai, takeda}@inf.kyushu-u.ac.jp

ABSTRACT

We consider music classification problems. A typical machine learning approach is to use support vector machines with some kernels. This approach, however, does not seem to be successful enough for classifying music data in our experiments. In this paper, we follow an alternative approach. We employ a (dis)similarity-based learning framework proposed by Wang et al. This (dis)similarity-based approach has a theoretical guarantee that one can obtain accurate classifiers using (dis)similarity measures under a natural assumption. We demonstrate the effectiveness of our approach in computational experiments using Japanese MIDI data.

1. INTRODUCTION

Music classification is an important problem in information retrieval from music data. There are a lot of researches to tackle the problem (see, e.g., [1,3,4,10,11,14,18]), as highly accurate music classifiers are useful for music search and feature extraction.

One of typical approaches to classify music is to represent each music data as a feature vector, which is then classified by standard machine learning methods. On the other hand, finding good features for music classification is a non-trivial task. For example, performance worm [15], performance alphabet [16], and other approaches including [1, 10, 11, 18].

Another popular approach in Machine Learning is to use support vector machines (SVMs) with kernels [7–9, 12, 19]. One way to improve accuracy of music classification is to design a good kernel for music data. This approach, however, does not seem to be very successful so far. As we will show later, well known string kernels such as n-gram kernels [12] and mismatch kernels [8] for texts do not obtain satisfactory results for music classification in our experiments. Further, to design a kernel, the function to be designed needs to be positive semidefinite, which is a limitation when we try to exploit the structure of music to improve classification accuracy.

In this paper, we follow an alternative approach. We employ a (dis)similarity-based learning framework proposed by Wang et al. [20]. This (dis)similarity-based approach has a theoretical guarantee that one can obtain accurate classifiers using (dis)similarity measures under a natural assumption. In addition, the advantage of this approach is able to use *any* (dis)similarity measures which do not have to be positive semidefinite and *any* data.

Further, we combine this (dis)similarity-based learning approach with 1-norm soft margin optimization formulation [5, 22]. An advantage of the formulation is that it is useful for feature selection because of the sparse nature of the underlying solution. In other words, the formulation help us to find "relevant" instances (i.e., music data) to classify music. Such relevant instances might contain representative features of the class. Therefore, it might be useful to extract good features.

For simplicity, throughout the paper, we deal with classification problems of symbolic music data such as MIDI files only. Thus we do not consider audio signal data and we assume (dis)similarity functions over texts. Note that our framework using (dis)similarity functions does not depend on the data format. We can deal with audio signal data as well if we employ (dis)similarity functions over signals.

We demonstrate the effectiveness of our approach in computational experiments using Japanese music data. Our approach, combined with non-positive semidefinite (dis)similarity measures such as edit distance, shows better performance than SVMs with string kernels.

2. LEARNING FRAMEWORK USING DISSIMILARITY FUNCTION

In this section, we review a learning framework using dissimilarity function proposed by Wang et al. [20]. Let X be the instance space. We assume that a dissimilarity function $d(x, x')$ is a function from $\mathrm{X} \times \mathrm{X}$ to $\mathbb{R}^+$. A pair (x, y) of instance $x \in \mathrm{X}$ and label $y \in \{-1, 1\}$ is called an *example*. For instance, X might be some set of MIDI data and then

an example is a pair of a MIDI file and positive or negative label. The learner is given a set S of examples, where each example is drawn randomly and independently from an unknown distribution P over $\mathrm{X} \times \{-1, +1\}$. Then, the learner is supposed to output a hypothesis $h(x) : \mathrm{X} \to \{-1, 1\}$. The goal of the learner is to minimize the error of the hypothesis h w. r. t. the distribution P, i.e., the probability that h misclassifies the label of a randomly drawn example (x, y) according to P, $\Pr_{(x,y)\sim P}(h(x) \neq y)$. In particular, we assume that a hypothesis is constructed using a dissimilarity function d. Also, we will use the notation that $\mathrm{sgn}[a] = 1$ if $a > 0$ and -1 otherwise.

Then we show a definition of "good" dissimilarity function.

Definition 1 (Strong (ϵ, η)-goodness, Wang et al. [20]) *A dissimilarity function $d(x, x')$ is said to be strongly (ϵ,η)-good, if at least $1-\epsilon$ probability mass of examples z satisfy:*

$$\Pr_{z',z''\sim P}(d(x, x') < d(x, x'')|y' = y, y'' = -y) \geq 1/2+\eta/2 \tag{1}$$

where the probability is over random examples $z' = (x', y')$ and $z'' = (x'', y'')$.

Roughly speaking, this definition says that for the most of random examples $z = (x, y)$ and random positive and negative examples, the instance x is likely to be closer to the instance with the same label. Then, under the natural assumption that the given dissimilarity function d is (ϵ, η)-good, we can construct an accurate classifier based on d, as is shown in the following theorem.

Theorem 1 (Wang et al. [20]) *If d is a strongly (ϵ, η)-good dissimilarity function, then with probability at least $1 - \delta$ over the choice of $m = (4/\eta^2)\ln(1/\delta)$ pairs of examples (z', z'') with labels $y' = 1, y'' = -1, i = 1, 2, ..., m$, the following classifier $F(x) = \mathrm{sgn}[f(x)]$ where*

$$f(x) = \frac{1}{m}\sum_{i=1}^{n} \mathrm{sgn}[d(x, x_i'') - d(x, x_i')]$$

has an error rate of no more than $\epsilon + \delta$. That is

$$\Pr_{z\sim P}(F(x) \neq y) = \Pr_{z\sim P}(yf(x) \leq 0) \leq \epsilon + \delta.$$

This theorem says that an unweighted voting classifier consisting of sufficiently many randomly drawn examples is accurate enough with high probability. We should note that the existence of a (ϵ, η)-good dissimilarity function might be too restrictive in some cases. For such cases, Wang et al. also proposed more relaxed definitions of good dissimilarity functions. Under such relaxed definitions, it can be shown that there exists a weighted combination

$$f(x) = \sum_{i=1}^{m} w_i h_i(x),$$

where each $w_i \geq 0$, $\sum_i w_i = 1$, $h_i(x) = \mathrm{sgn}[d(x_i'', x) - d(x_i', x)]$ and x_i'' and x_i' are positive and negative instances, such that $\mathrm{sgn}[f(x)]$ is accurate enough (see [20] for the details).

3. OUR FORMULATION

In this section, we consider how to find an accurate weighted combination of base classifiers consisting of a pair of positive and negative instances. To do so, we employ the 1-norm soft margin optimization, which is a standard formulation of classification problems in Machine Learning (see,e.g, [5, 21]). Simply put, the problem is to find a linear combination of base classifiers (or a hyperplane over the space defined by base classifiers) which has large margin with respect to examples, where the margin of a linear combination w with respect to an example z is a distance between w and z. In fact, the large margin generalization theory (e.g., [17]) guarantees that a weighted combination of base classifier is likely to have higher accuracy when it has larger margin w.r.t. examples. Further, an additional advantage of 1-norm soft margin optimization is that the resulting linear combination of base classifiers is likely to be sparse since we regularize 1-norm of the weight vector. This property is useful for feature selection tasks.

3.1 The 1-norm soft margin formulation

Suppose that we are given a set $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, where each (x_i, y_i) is an example in $\mathrm{X} \times \{-1, +1\}$. Here, following the dissimilarity-based approach in the previous section, we assume the set of hypotheses, $H = \{h(x) = \mathrm{sgn}[d(x_i, x) - d(x_j, x)] \mid x_i \text{ and } x_j \text{ are positive and negative instances in } S, \text{ respectively } \}$. For simplicity of the notation, we denote H as $H = \{h_1, \ldots, h_n\}$, where n is the number of pairs of positive and negative examples in S. Then, the 1-norm soft margin optimization problem is formulated as follows (e.g. [5, 21]):

$$\max_{\rho,b\in\mathbb{R},\boldsymbol{w}\in\mathbb{R}^n,\boldsymbol{\xi}\in\mathbb{R}^m} \rho - \frac{1}{\nu}\sum_{i=1}^{m}\xi_i \tag{2}$$

sub.to

$$y_i(\sum_j w_j h_j(x_i) + b) \geq \rho - \xi_i (i = 1, \ldots, m),$$

$$\boldsymbol{w} \geq \mathbf{0}, \sum_{j=1}^{n} w_j = 1$$

$$\boldsymbol{\xi} \geq \mathbf{0}.$$

Here the term $y_i(\sum_j w_j h_j(x_i)+b)$ represents the margin of the hyperplane $(\boldsymbol{w}, b)$ w.r.t. an example (x_i, y_i) when the 1-norm of w is constrained to be 1. It is known that the margin is measured as ∞-norm distance between $(\boldsymbol{w}, b)$ and

(x_i, y_i) [13]. The parameter ρ means the minimum margin. Note that if the margin is positive w.r.t. all the examples, the examples are linearly-separable. For the case when the data is inseparable, we allow each example to violate the minimum margin ρ by the amount of ξ_i. So, the problem is to maximize the minimum margin ρ while minimizing the sum of losses defined as ξ_i. The parameter $\nu \in [1, m]$ controls the tradeoff between maximization of the margin and minimization of losses.

By using Lagrangian duality (e.g. [2]), the dual problem is given as follows:

$$\begin{aligned} &\min_{\gamma, \boldsymbol{d}} \gamma \qquad (3)\\ &\text{sub.to}\\ &\text{Edge}_{\boldsymbol{d}}(h_j) = \sum_i d_i y_i h j(x_i) \leq \gamma (j = 1, \ldots, n),\\ &\boldsymbol{d} \leq \frac{1}{\nu}\mathbf{1},\\ &\boldsymbol{d} \geq \mathbf{0}, \sum_{i=1}^{m} d_i = 1\\ &\boldsymbol{d} \cdot \boldsymbol{y} = 0. \end{aligned}$$

The dual problem is about finding a distribution $\boldsymbol{d}$ over examples satisfying linear constraints. In particular, since $y_i h_j(x_i) = 1$ if and only if $h_j(x_i) = y_i$, $\text{Edge}_{\boldsymbol{d}}(h_j)$ can be viewed as a weighted accuracy of the hypothesis of h_j w.r.t. the distribution $\boldsymbol{d}$. So, in other words, a solution $\boldsymbol{d}^*$ of the dual problem is the most "difficult" distribution w.r.t. hypotheses in H. Note that, since the both problems (2) and (3) are linear programs, these problems are equivalent. That is, if we solve one problem, we can obtain a solution of the other problem as well.

We solve the dual problem (3) using LPBoost [5], which is shown in Algorithm 1. LPBoost chooses a hypothesis $h \in H$ and solve a sub-problem of the dual problem (3) iteratively until some termination condition is satisfied. It is known that after sufficient number of iterations, output by LPBoost converges to a solution of the problem (3). More precisely, the following statement holds for any given precision parameter $\lambda > 0$.

Theorem 2 (Demiriz et al. [5]) *LPBoost outputs a final hypothesis such that the corresponding solution $(\gamma_T, \boldsymbol{d}_T)$ satisfies $\gamma_T \leq \gamma^* + \lambda$, where $(\gamma^*, \boldsymbol{d}^*)$ is an optimal solution of the dual problem (3).*

4. COMPUTATIONAL EXPERIMENT

In this section, we show preliminary experimental results. The task we consider is classification problems over a data set of Japanese songs.

Algorithm 1 LPBoost(S,λ)

(1) Let $\boldsymbol{d}_1$ be the uniform distribution over S.

(2) For $t = 1, \ldots,$

(a) Choose a hypothesis $h^{(t)} \in H$ whose edge w.r.t. $\boldsymbol{d}_t$ is more than $\gamma_t + \lambda$.

(b) If such a hypothesis does not exist in H, let $T = t - 1$ and break.

(c) Solve the soft margin optimization problem (3) w.r.t. the restricted hypothesis set $\{h^{(1)}, \ldots, h^{(t)}\}$. Let $(\gamma_{t+1}, \boldsymbol{d}_{t+1})$ be a solution.

$$\begin{aligned} (\gamma_{t+1}, \boldsymbol{d}_{t+1}) = \arg\min_{\gamma, \boldsymbol{d}} &\gamma\\ \text{sub. to}&\\ &\sum_i d_i y_j h^{(j)}(x_i) \leq \gamma \ \ (j = 1, \ldots, t)\\ &\boldsymbol{d} \leq \frac{1}{\nu}\mathbf{1}. \end{aligned}$$

(3) Output $f(x) = \sum_{t=1}^{T} w_t h^{(t)}(x)$, where each $\boldsymbol{w}_t$ ($t = 1, \ldots, T$) is a Lagrange dual of the soft margin optimization problem (3).

4.1 Data set

Our data set of Japanese songs consists of 119 pop songs (JPOP) and 119 Enka songs, where Enka is a genre of Japanese songs whose style is rather close to traditional folklore songs. We convert MIDI format into string data according to the method specified in Kadota et al. [6].

For the original data in the MIDI format, we specify a particular channel which corresponds to principal melody, and extract a single sequence consisting of notes and rests, where a note is a pair of pitch and duration values and a rest has only a duration value. We choose the highest pitch if more than one pitch is "NOTE ON" at an instant. In addition we quantize the obtained data so that all the duration values are multiples of the MIDI delta time correspinding to the sixteenth note. Then we convert the quantized note/rest sequences into string data of three types (see Figure 1):

Pitch string We divide each note (rest) into sixteenth notes (rests) to produce a string consisting of pitches and rests. For simplicity, we ignore an octave difference, and therefore the number of possible pitches is twelve. The alphabet size is thus 13.

Rhythm string Similarly, we divide each note (rest) into sixteenth notes (rests) and produce a string consisting of four symbols: N (beginning fragment of a note),

Classifier		SVM		our method			
(dis)similarity measure		n-gram kernel	mismatch kernel	n-gram kernel	mismatch kernel	edit distance	LCS
Pitch	Nontransposed	61.34	65.55	70.16	73.52	**86.12**	79.42
	Transposed	61.34	65.55	70.16	73.52	**86.12**	79.42
Rhythm		86.97	86.97	88.67	89.90	87.79	**92.87**
Note	Nontransposed	66.39	71.01	76.46	79.81	**87.38**	85.33
	Transposed	66.39	71.01	76.46	79.81	**87.38**	85.33

Table 1. Classification accuracy (%)

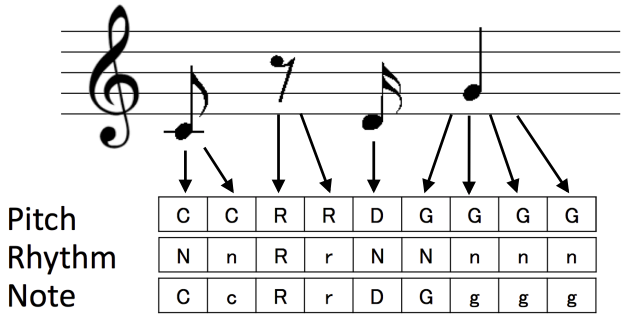

Figure 1. How to extract string data from note/rest sequence.

n (non-beginning fragment of a note), R (beginning fragment of a rest) and r (non-beginning fragment of a rest).

Note string Composition of pitch and rhythm strings. That is, from pitch string $a_1 \dots a_m$ and rhythm string $b_1 \dots b_m$ for a same note/rest sequence, we composed the string $(a_1, b_1) \dots (a_m, b_m)$. The alphabet size is 26.

For pitch strings and note strings, we have an option to transpose them into C major (C minor).

4.2 Classification algorithms

The algorithms we examined are SVMs with string kernels and LPBoost with the (dis)similarity-based learning framework (our method). For SVMs, we used n-gram kernels [9] with $n = 1, \dots, 10$ and mismatch kernels [8] with parameters $n = 2, \dots, 20$ and $k = 1, \dots, n-1$. For other settings, We used default parameters of LIBSVM for SVMs.

For our method, we used two (dis)similarity measures: the length of Longest Common Subsequence (LCS) and the edit distance, in addition to the string kernels used for SVMs. For the parameter ν, we set $\nu = cm$, where m is the given sample size and $c = 0.05, 0.1, 0.15, 0.2, 0.25.0.3$. As described in Section 3, we used base classifiers $h(x) = \mathrm{sgn}[d(x_i, x) - d(x_j, x)]$ associated with pairs of positive instance x_i and negative instance x_j. In total, we used $119 * 119 = 14161$ base classifiers.

We evaluated SVMs and our method by performing 5-fold cross validation. The results are summarized in Table 1, where the accuracies of respective methods are shown with best parameters.

4.3 Result

As is shown in Table 1, our method shows better performance than SVMs with all kernels. For pitch string and note string, the best value was obtained by our method with the edit distance. For rhythm string, the best value was gained by our method with LCS. For pitch string and note string, transposition in the note did not affect the classification accuracy in our experiments.

Our methods with the edit distance and with LCS have better results than those with the n-gram and the mismatch kernels. This might be because the edit distance and LCS capture characteristics of JPOP and Enka better. Over all of the (dis)similarity measures and kernels we used, the best classification results were obtained on rhythm string. This might be because JPOP has rather high tempo while Enka has slow tempo.

Finally, we investigate which base classifiers

$$h(x) = \mathrm{sgn}[d(x_i, x) - d(x_j, x)],$$

associated with pairs of JPOP x_i and Enka x_j, contribute an accurate classification.

In the case of our method with the edit distance on rhythm strings, among all possible 14,161 pairs, at most 66 pairs have a non-zero weight in the final weighted combination for all the parameters c. So, the obtained weighted combination is quite sparse.

We observe that the resulting final weighted combination is sparser when we employ (dis)similarity measures.

For rhythm string, we choose $c = 0.3$ and $c = 0.15$ which give the best classification results for LCS and the edit distance, respectively. We arrange all the pairs in decreasing order of their weights, and the top 10 pairs are displayed in Tables 2 and 3.

c	JPOP Title	ENKA Title	Weight	Total of weight
0.3	Secret Heaven	Norennohana	0.430940	0.43094
	Secret Heaven	Aiha Kirameite	0.146408	0.577349
	In My Room	Akashiabanka	0.140884	0.718233
	Kimiga Suki	Amagigoe	0.135359	0.853591
	Raven	Nyonin Kouya	0.116022	0.969613
	Raven	Okuhidabojou	0.024862	0.994475
	Kimiga Suki	Unga	0.005525	1

Table 2. Top 10 pairs with large weight in the final weighted combination for the edit distance.

c	JPOP Title	ENKA Title	Weight	Total of weight
0.15	Tsukiyo no koibitotachi	Ohsakawan	0.208197	0.208197
	Amenimo Makezu	Kaettekoiyo	0.127717	0.335914
	Totsuzen	Otokogi	0.059798	0.395712
	Only You	Matsuri	0.054587	0.450299
	Totsuzen	Shiroi Yuki	0.050715	0.501013
	FINAL DISTANCE	Yukimoete	0.050270	0.551284
	Tsukiyo no koibitotachi	Hashi	0.046641	0.597925
	Goodbye Yesterday	Yoshida Shoin	0.044228	0.642153
	Secret Heaven	Kokoha Minatomachi	0.039791	0.681944
	FINAL DISTANCE	Ettou Tsubame	0.037098	0.719042

Table 3. Top 10 pairs with large weight in the final weighted combination for LCS.

In the case of the edit distance, only the top 3 pairs occupy more than 70% of total weight, and the top 5 pairs occupy more than 90% of total weight. We omitted the last three pairs in the top 10 list of Table 2 since their weights are less than 10^{-17}. So, only at most 5 pairs of JPOP and Enka contribute the final classification significantly. Similarly, in the case of LCS, the top 10 pairs have about 70% of total weight. These songs in the top lists might be "representatives" of JPOP or Enka, from which we might be able to extract good feature representations.

5. CONCLUSION

In this paper we addressed the music classification problem. We employed the (dis)similarity-based learning framework proposed by Wang et al. [20]. Computational experiments show that our method combined with string kernels such as the n-gram and the mismatch kernels outperform SVM with them. One advantage of our approach is that it can be used combined with *any* (dis)similarity measure, which do not have to be positive semidefinite. In fact, our method with LCS and the edit distance show better classification accuracy than with the string kernels. Among the three types of string data we examined, the rhythm string seems most suited for genre classification in our experiments. Songs in the pairs with large weight in the resulting weighted combination might be representatives of respective music genres. We challenge classification problem with data set of 238 songs, however, the data set is too low to be generality of this approach. We need to experiment bigger amounts of data, and we measure classification accuracy of not only symbolic data but also audio data. Future work is not only music genre classification but also automatic extraction of features of music genres or composers.

6. REFERENCES

[1] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65:473–484, 2006.

[2] Stephen Boyd and Lieven Vandenberghe, editors. *Convex Optimization*. Cambridge University Press, 2004.

[3] Rudi Cilibrasi, Paul Vitányi, and Ronald de Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.

[4] Christopher DeCoro, Zafer Barutcuoglu, and Rebecca Fiebrink. Bayesian aggregation for hierarchical genre classification. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07)*, 2007.

[5] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002.

[6] Takashi Kadota, Masahiro Hirao, Akira Ishino, Masayuki Takeda, Ayumi Shinohara, and Fumihiro Matsuo. Musical sequence comparison for melodic and rhythmic similarities. In *Proc. 8th International Symposium on String Processing and Information Retrieval (SPIRE '01)*, pages 111–122, 2001.

[7] Chiristina Leslie and Rui Kang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 5:1435–1455, 2004.

[8] Christina S. Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston, and William Stafford Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4):467–476, 2004.

[9] Christina S. Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proc. the Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[10] Thomas Lidy and Andreas Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05)*, pages 34–41, 2005.

[11] Thomas Lidy, Andreas Rauber, Antonio Pertusa, and José Manuel Iñesta. Improving genre classification by combination of audio and symbolic descriptors using a transcription system. In *Proceedings of 8th International Conference on Music Information Retrieval (ISMIR'07)*, 2007.

[12] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, Chris Watkins, and Bernhard Scholkopf. Text classification using string kernels. *Journal of Machine Learning Research*, 2:563–569, 2002.

[13] O. L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24:15–23, 1999.

[14] Carlos Pérez-Sancho, David Rizo, and José Manuel Iñesta. Genre classification using chords and stochastic language models. *Connection Science*, 21:145–159, 2009.

[15] P.Zanon and G.Widmer. Learning to recognize famous pianists with machine learning techniques. In *Proceedings of the Stockholm Music Acoustics Conference*, 2003.

[16] Craig Saunders, David R. Hardoon, John Shawe-taylor, and Gerhard Widmer. Using string kernels to identify famous performers from their playing style. In *Proceedings of the 15th European Conference on Machine Learning*, pages 384–395, 2004.

[17] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wen S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

[18] George Tzanetakis, Georg Essl, and Perry Cook. Automatic musical genre classification of audio signals. In *Proceedings of the 2nd International Symposium on Music Information Retrieval (ISMIR' 01)*, 2001.

[19] S. V. N. Vishwanathan and A. J. Smola. Fast kernels on stirngs and trees. In *Advances on Neural Information Proccessing Systems 14*, 2002.

[20] Liwei Wang, Masashi Sugiyama, Cheng Yang, Kohei Hatano, and Jufu Feng. Theory and algorithm for learning with dissimilarity functions. *Neural Computation*, 21:1459–1484, 2009.

[21] M. Warmuth, K. Glocer, and G. Rätsch. Boosting algorithms for maximizing the soft margin. In *Advances in Neural Information Processing Systems 20*, pages 1585–1592, 2008.

[22] Manfred K. Warmuth, Karen A. Glocer, and S. V. N. Vishwanathan. Entropy regularized lpboost. In *Proceedings of the 19th international conference on Algorithmic Learning Theory (ALT '08)*, pages 256–271, 2008.